Debugging CUDA Applications with Allinea DDT

Ian Lumb
Sr. Systems Engineer, Allinea Software Inc.
ilumb@allinea.com

GTC 2013, San Jose, March 20, 2013
Challenges for Developers

Scale
Heterogeneity
Diversity
Legacy
Embracing GPUs

- GPUs – a rival to traditional processors
  - Great price/performance ratios
- New languages, compilers, standards
  - CUDA, OpenACC, OpenCL, ...
- HPC developers need to consider
  - Data transfer
  - Multiple memory levels
  - Grid/block layout and thread scheduling
  - Synchronization
- Bugs are inevitable
The Allinea Environment: Benefits

- **At last:** a modern **integrated** environment for the HPC developer

- Supporting the lifecycle of application development and improvement
  - Productively debug code
  - Enhance application performance

- Designed for productivity
  - Consistent integrated easy to use tools
  - Enables effective HPC development

- Improve system usage
  - Fewer failed jobs
  - Higher application performance
World-class Parallel Debugging

- Full-scale parallel debugging
- Full CUDA 5 support
- OpenMP + OpenACC
- Mixed MPI / OpenMP / CUDA
- Debug the real device

www.allinea.com
Allinea DDT Demo:
CUDA Debugging Essentials
__global__ void permute(int n, int *data) {
    __shared__ int smem[256];
    if (n <= 1)
        return;
    smem[threadIdx.x] = data[threadIdx.x];
    __syncthreads();
    smem[threadIdx.x] *= 2; // permute data in a trivial way
    __syncthreads();
    // Write back to GMEM since we can’t pass SMEM to children.
    data[threadIdx.x] = smem[threadIdx.x];
    __syncthreads();
    if (threadIdx.x == 0) {
        permute<<<1, 256>>>(n/2, data);
        permute<<<1, 256>>>(n/2, data+n/2);
        cudaDeviceSynchronize();
    }
}
```c
#include <stdio.h>

__global__ void permute(int n, int *data) {
    __shared__ int smem[256];
    if (n <= 1 || threadIdx.x >= n)
        return;
    smem[threadIdx.x] = data[threadIdx.x];
    __syncthreads();
    smem[threadIdx.x] *= 2; // permute data in a trivial way
    __syncthreads();
    // Write back to GEMM since we can't pass SMEM to children.
    data[threadIdx.x] = smem[threadIdx.x];
    __syncthreads();
    if (threadIdx.x == 0) {
        permute<<<1, 256, 0>>>(n/2, data);
        permute<<<1, 256, 0>>>(n/2, data+n/2);
    }
}

int main(int argc, char** argv) {
```

OpenACC Debugging

- On device debugging with Allinea DDT
  - Variables – arrays, pointers, full F90 and C support
  - Set breakpoints and step warps and blocks
- Identical to CUDA
  - Full warp/block/kernel controls
  - CPU vs GPU distinction still present
- Status
  - Allinea supports on-device debugging for Cray compiler
  - Other compilers to follow
World-class Parallel Profiling

- Full-scale parallel profiling
- Just 5% runtime overhead
- No need to instrument code
- MPI, memory and CPU
- See the slowest lines of code

www.allinea.com
Use of Allinea DDT at SciNet

- Productively **debug** parallel CUDA code
- Completely **understand** parallel CUDA code
  - Interact with data, algorithms, codes, programs and applications in real time
- **Develop** parallel CUDA code from scratch
- **Port** parallel algorithms, codes, programs and applications to CUDA
- **Scale** CUDA algorithms, codes, programs and applications

http://www.scinethpc.ca/
Petascale Debugging

- Cray XK* systems
  - ORNL Titan
    - MPI debugging proven at 220,000 CPU cores in 2010
      - ~300,000 CPU cores in 2012
    - MPI-OpenACC hybrid codes expected to scale similarly
  - NCSA Blue Waters
    - ~700,000 CPU cores
Allinea at GTC 2013

- Please visit our booth!!!
  - Live demonstrations
  - Further information

- Call to action: Evaluate Allinea tools!
  http://www.allinea.com/products/
The Allinea Environment - an integrated, ready-to-run development suite

Get correct results fast using the industry-leading parallel debugger

Full support for CUDA 5

See which loops can be offloaded to the GPU most effectively with Allinea MAP
allinea

Leaders in parallel software development tools
Allinea DDT: Fixing bugs made easy

- A tool that allows you to solve your problems faster
  - Control threads and processes en-masse
  - Syntax-highlighting source browser
  - Parallel stacks and variable views that highlight divergence
- Supports the latest in MPI, OpenMP, CUDA and more
  - CUDA 5.0 and Kepler K20
  - Intel Xeon Phi coprocessor
Allinea DDT: Proven to the extreme

- Scalability by design
  - User interface that scales
  - High performance tree architecture
- Proven performance at Petascale
  - Measured in milliseconds
  - Routine use at 100,000+ cores
Allinea DDT: More than debugger

- Integrated automated detection of bugs
  - Static analysis
  - Memory leaks and errors
- Open plugin architecture
  - MPI checking tools
- Offline mode - debug in batch mode

```c
threads = calloc(sizeof(pthread_t), nthreads);
ids = calloc(sizeof(int), nthreads);
init_mutex();
pthread_mutex_lock(mutley);
for (i = 0; i < nthreads; ++i) {
    ids[i] = i;
    pthread_create (threads + i, NULL, &thread,
    }
pthread_mutex_unlock(mutley);
for (i = 0; i < nthreads; ++i)
    pthread_join (threads[i], NULL);

return 0;
```
allinea
Leaders in parallel software development tools
Allinea DDT 3.2.1 (10/2012)

- CUDA 5.0 support
- Intel Xeon Phi support
- MPMD support
  - MPICH 2
  - Intel MPI
- IBM BlueGene /Q port
Allinea Tools 4.0 (3/2013)

- GA introduction of Allinea MAP
  - Observe and improve the performance of your MPI application
- Improved syntax highlighting and code folding in code viewer
- Remote client
  - Native client for Linux, Windows and Mac
- Redesigned welcome page
- Thread viewer
  - Shows OpenMP thread ID for supported implementations
Allinea Tools 4.0 (2)

- Support for MPICH 3
- Cray-specific
  - Attach to GPU codes (Cray XK7)
  - Memory Debugging Library preloading
    - C/Fortran (no threads/threads) now supported (Cray XK6/7)
    - C++ (no threads/threads) not supported
A Unified Environment for HPC

Shared Graphical Interface

Shared Configuration Files

Shared Scalable Architecture

Intelligent data consolidation

allinea
www.allinea.com
Allinea MAP: The profiler you'll want to use

- Works first time, every time
- From one process to tens of thousands
  - 5% slowdown
  - No instrumentation and no huge data files
- No need to recompile
Allinea MAP: Refreshing simple

- See where time is spent in your source code.
- Visualize the entire run
- Zoom in to explore iterations, functions and loops.
Allinea MAP: Surprisingly deep

- A wide range of metrics for deep insights into performance:
  - CPU: Time spent in FPU, vector (SSE/AVX) and memory operations
  - MPI: Time spent and bytes transferred in P2P / collective operations
- Visual display of distribution over ranks, min / max ranks labelled
- Export data via XML or CSV for deeper analysis and reports
Allinea MAP: Built on strength

- World-class scalability
  - Shares Allinea DDT tree architecture – proven beyond Petascale
  - Data is merged on the cluster: no huge files.